REMARKS

Claims 3, 4, 10, 11, 17, 18, and 22 to 35 were pending in the application at the time of the final examination. Claims 3, 4, 10, 11, 17, 18, and 22 to 35 remain rejected as obvious.

Applicant has amended each of Claims 3, 10, 17, 22, 24, 27, 30, and 33 to make explicit that which was implicit in the Claims. In particular, the public classes and interfaces are explicitly recited as being written in an object-orientated programming language. This distinguishes over an object that is an instance of a class, and structures in non-object oriented programming languages. This amendment was inherent in the original claim language and is being made in an attempt to move prosecution forward and avoid any discussion on whether Applicant is requesting the Examiner to read limitations from the specification into the claims.

The First Obviousness Rejection is not Well Founded

Claims 3, 4, 10, 11, 17, 18, 22 to 25, 27, 28, 30, 31, 33, and 34 stand rejected under 35 U.S.C. § 103 as being unpatentable over U.S. Patent No. 5,408,665, hereinafter referred to as Fitzgerald, in view of U.S. Patent No. 6,526,571, hereafter referred to as Aizikowitz, and U.S. Patent No. 5,907,704, hereinafter referred to as Gudmundson.

Applicant respectfully notes that the MPEP requires for a combination of references that:

> If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims *prima facie* obvious.

MPEP § 2143.01, 8th Ed., Rev. 3, p. 2100-138 (August 2005)

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

Here, the combination mixes concepts from different programming levels and for different features to arrive at Applicant's invention as recited in Claim 3. Moreover, the proposed modification changes the principles of operation of the primary reference. Accordingly, the analysis is not piecemeal as alleged, but rather a demonstration that the obviousness rejection is not well founded.

First, Fitzgerald is citing a teaching creation of a standard dictionary that includes a list of the library's public symbols and module names. The rejection then stated:

> Fitzgerald does not explicitly state that the library (260) is an object-orientated library as claimed, and then does not expressly state that the public elements art public classes and interfaces. However, Fitzgerald does state that in the preferred embodiment, the programming language specific to the system is Borland C++.

Fitzgerald is careful to distinguish between the compiler and the object modules in the library and the computer programming language used in both. Fitzgerald stated:

> In a preferred embodiment, the compiler 220 is Borland C++ compiler.

Fitzgerald, Col. 5, lines 46, 47.

Thus, it is the compiler that is a C++ compiler, which implies that source listings 120 of Fitzgerald may be object-orientated. However, Fitzgerald expressly teaches that the library and the object modules in the library are not written in an object-orientated language. Specifically, Fitzgerald taught:

> During creation of a program, individual translator outputs (i.e., outputs from compiler and assemblers) are linked together to create the executable program. The translator outputs or object modules ("OBJs") store a plurality of records describing the 80x86 object language

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

> used for input and output of object language processors, such as linkers and librarians. The order of the records is to some extent arbitrary.

Fitzgerald, Col. 5, line 63 to Col. 6, line 2.

Thus, Fitzgerald taught that the compiler output is a translator output and then that the records in that output are in the 80X86 object language which is assembly language for the Intel 80X86 processor family. It is well known that the 80X86 object language is not an object-orientated language. Thus, the rejection confuses the input to the compiler and the output from the compiler. The library is associated with the output of the compiler and is expressly taught as being written in a language that is not object-oriented.

Further, the Fitzgerald expressly defines the Standard Dictionary that was cited as teaching the public list of Claim 3. First, Fitzgerald stated:

> Object (OBJ) files may be placed by the Librarian 265 (of FIG. 2A) in a single "library" (.LIB) file for collective storage. An exemplary library format is that provided by Microsoft for MS-DOS. In general layout, a Microsoft MS-DOS library file is constructed in a manner similar to that of the above-described Intel object module--it comprises a plurality of records, most of which will be one of the aforementioned object record types. As shown in FIG. 3C, a library 350 essentially comprises a collection of object modules 370 followed by a Standard Dictionary 360.

Fitzgerald, Col. 8, lines 15 to 25.

Thus, Fitzgerald teaches that the Librarian that was cited as creating the public list generates a single file for MS-DOS. MS-DOS is not an object-oriented operating system and so this is but further evidence that the librarian is not at the class and interface level as recited in Claim 3. Further,

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

> Standard Dictionary 430 of Library 410 is modified from the Microsoft standard for the MD-DOS library format.

Fitzgerald, Col. 10, lines 11 to 13.

Thus, the standard dictionary is taught as being a modified from of the MS-DOS library format. Next, the entries in the standard dictionary are expressly taught, specifically,

> Entries of the dictionary 430 are structured as follows. The first byte stores the length of the symbol to follow; this is followed by the bytes of text for the symbol. Next, a word field (two bytes in byte-swapped order) specifies the page number at which the module defining the symbol begins; the library header is counted as the 0th page.
>
> Following the page number word field there are either two or three additional bytes of information generated by the librarian 265 to support Extended Dictionary processing. The additional information in the Standard Dictionary does not strictly conform to the Microsoft standard for the MD-DOS library format. Since symbols in the Standard Dictionary are obtained by a random access method using a page address scheme (block or page/bucket index), the information is essentially "hidden". The additional data bytes stored at the end of the symbol are not detected by other manufacturer's librarian tools.
>
> The additional data bytes function as follows. If the symbol is a public symbol, the next three bytes are essentially a pointer into the Standard Dictionary to the name of the module which defines the public. The first two bytes are a byte-swapped word field that specifies the page number (index to a standard dictionary block) and the next byte is a bucket index (pointer to symbol string in the standard dictionary block); thus, this data functions essentially as a pointer to the module name of the module that defines the public symbol. If the symbol is a module name, the next two bytes are a module number. This data is essentially an index (via the Extended Dictionary Index) into the Dependency Lists and Unresolved Externals List for the module.

Fitzgerald, Col. 10, lines 13 to 46.

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

Thus, Fitzgerald expressly teaches that the Standard Dictionary functions at the assembly code level. Nevertheless, the rejection ignores this explicit teachings and asserts that one of skill in the art would look to Aizikowitz and then backfit Fitzgerald based on a class hierarchy. This is error at multiple levels.

First, Fitzgerald, as one of skill in the art, expressly contemplates processing object-orientated source programs with the C++ complier. Fitzgerald then describes a system that works and builds a library of object modules in an assembly level language. Fitzgerald did not consider a list as recited in Claim 1 and the rejection has failed to establish why Fitzgerald would need such a list when it provides a working system. Nevertheless, the rejection asserts that one of skill in the art would use the teachings of Aizikowitz and go into Fitzgerald and modify the public symbol names of Fitzgerald to public symbols and public modules to classes and interfaces.

However, the need for such a change not been demonstrated, the rejection stated "One would have been motivated to do so because of the suggestions provided by Fitzgerald as above. Fitzgerald does not provide a motivation to change Fitzgerald, because Fitzgerald worked for its intended purpose. Moreover, Fitzgerald considered the information and the rejection admits that Fitzgerald did not incorporate the feature. Accordingly, Fitzgerald not only does not suggest the modification, but also teaches away from the modification when Fitzgerald considered that very information cited as the motivation for the combination.

Finally, a further modification would be made based on the third reference to exclude classes that were not public, but again, it has not been shown how Fitzgerald would work for its intended purpose after such a modification. If a non-public class is needed to create an executable object module and the class is excluded, the object module will not execute. The

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

MPEP is clear that the primary reference must still work for its intended purpose after the modifications. The purpose of Fitzgerald was to generate a library that could be used in executable program code. Accordingly, the modification based on the third reference goes against the purpose of Fitzgerald. Applicant incorporates herein by reference the prior remarks directed at the motivation to combine the references. Applicant requests reconsideration and withdrawal of the first obviousness rejection of Claim 3.

Claim 4 depends from Claim 3 and so distinguishes over the combination of references for at least the same reasons as Claim 3. Applicant requests reconsideration and withdrawal of the first obviousness rejection of Claim 4.

Independent Claims 10, 17, and 22 include the limitations discussed above with respect to Claim 3 and so the comments with respect to Claim 3 are applicable and incorporated herein by reference. Applicant requests reconsideration and withdrawal of the first obviousness rejection of each of Claims 10, 17, and 22.

Claim 11 depends from Claim 10, Claim 18 depends from Claim 17, and Claim 23 depends from Claim 22, and so each of Claims 11, 18, and 23 distinguishes over the combination of references for at least the same reasons as the independent claim from which it depends. Applicant requests reconsideration and withdrawal of the first obviousness rejection of each of Claims 11, 18, and 23.

In the obviousness rejection of Claim 24, the rejection cited "Standard and Extended Dictionaries" as "an application programming interface definition file." However, in the rejection of Claim 3, the "Standard Dictionary" was cited as a "public list." The Examiner has failed to explain or establish that an API, which is a well-known term in the art, has anything to do with a dictionary used in linking computer program code as in Fitzgerald. Moreover, the Examiner has

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

cited no teaching that the dictionaries have any hierarchical nature or any sort of parent-child relationship. These characteristics must be added based solely on the secondary reference.

The rejection has not established how Fitzgerald would work for its intended purpose after the proposed modifications. Further inferences, as used in the rejection, are not teachings in the prior art references. For example, if an object module in Fitzgerald is a square root function or some other math function that is included in a mathematical library, what is the "direct parent of that module," and how would that be correlated with anything in the secondary reference.

In particular, Figs. 6A to 6D of Fitzgerald are "a method . . . for linking compiled object modules with Extended Dictionary Support." Linking compiled object modules and receiving a list of libraries by a linker is wholly unrelated to the process of Aizikowitz. As noted above with respect to Claim 3 and incorporated herein by reference, the combination of references requires multiple redefinitions and modifications to the primary reference that are not well founded. The fact, as originally noted, that the same object in Fitzgerald is being given different interpretations based upon the claim being rejected shows that the reference is not being consistently interpreted and considered as a whole. Applicant requests reconsideration and withdrawal of the first obviousness rejection of Claim 24.

Claim 25 depends from Claim 24 and so distinguishes over the combination of references for at least the same reasons as Claim 24. Applicant requests reconsideration and withdrawal of the first obviousness rejection of Claim 25.

Independent Claims 27, 30, and 33 include the limitations discussed above with respect to Claim 24 and so the comments with respect to Claim 24 are applicable and incorporated herein by reference. Applicant requests reconsideration and

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

withdrawal of the first obviousness rejection of each of Claims 27, 30, and 33.

Claim 28 depends from Claim 27, Claim 31 depends from Claim 30, and Claim 34 depends from Claim 33, and so each of Claims 28, 31, and 34 distinguishes over the combination of references for at least the same reasons as the independent claim from which it depends. Applicant requests reconsideration and withdrawal of the first obviousness rejection of each of Claims 28, 31, and 34.

The Second Obviousness Rejection is not Well Founded

Claims 26, 29, 32, and 35 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Fitzgerald in view of Aizikowitz and Gudmundson and further in view of U.S. Patent No. 5,974,255 hereinafter referred to as Gossain.

Applicant respectfully traverses the obviousness rejection of each of Claims 26, 29, 32, and 35 and incorporates herein by reference the above comments with respect to the independent claims that each depends. The additional information cited by the Examiner does not overcome the shortcomings of the primary combination. Applicant requests reconsideration and withdrawal of the second obviousness rejection of each of Claims 26, 29, 32, and 35.

The Third Obviousness Rejection is not Well Founded

Claims 3, 4, 10, 11, 17, 18, 22, and 23 stand rejected under 35 U.S.C. §103(a) as being allegedly unpatentable over U.S. Patent No. 6,230,314, hereinafter referred to as Sweeney, in view of Aizikowitz and Gudmundson.

Applicant respectfully traverses the obviousness rejection of Claim 3. Again, the combination of references proposes to modify the primary reference to work in a different way without

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

any motivation other than the original reference itself. The purpose of Sweeney was:

> . . . a class specialization technique is provided that eliminates redundant components from objects. More specifically, the technique detects situations where a member of a given class is used by some, but not all instances of that class, and eliminates this member from the instances where it is not needed. This is accomplished by an analysis of the program and its class hierarchy, followed by the construction of a new, specialized class hierarchy and a transformation of the program. These operations preserve the original behavior of the program, and have the effect of "optimizing away" unneeded class members from objects.

Sweeney, Col. 6, lines 16 to 28.

Thus, Sweeney is directed at removing redundant components from an object, which is an instantiation of a class. The citation to the general description of a class in Sweeney extracts only a piece of the reference and fails to consider the reference as a whole. Working within an object to remove redundant components is fundamentally different from "representing an application programming interface (API) definition for an object-oriented library" as recited in Claim 3.

Nevertheless, the rejection proposes to modify Sweeney to represent such an API, which in turn changes the principles of operation of Sweeney. There has been no citation to how Sweeney would determine redundant components in an object after such a modification or why Sweeney would even want to make such a change because Sweeney worked for its intended purpose. General background comments were clearly considered by Sweeney and a system was built that did not include such an API. Since Sweeney considered the very information used as a basis for modifying Sweeney, Sweeney teaches away from such a modification.

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

Extracting a piece of a method for removing redundant components, modifying that piece with a second reference and placing the modified piece back in Sweeney must result in a system that still works for the purpose intended by Sweeney. The rejection has not established how this would be done or even considered the issue.

Further, the information extracted from the secondary reference is being used in a way different from that taught by the reference. Again, a motivation must be provided for such a different application. Specifically, Aizikowitz stated:

> a method that belongs to a packaged (i.e. non-public) class or interface, cannot be directly overridden by a method from a different package; (Emphasis Added.)

Aizikowitz, Col. 4, lines 28 to 30.

Accordingly, Aizikowitz defines that a packaged method class or interface is non-public. With respect to Fig. 2, Aizikowitz stated:

> FIG. 2 shows pictorially an inheritance graph of a sealed package depicted generally as 10 having root interfaces I.sub.1, I.sub.2 and I.sub.4 and a root class C.sub.o.

Aizikowitz, Col. 5, lines 11 to 13.

Thus, Fig. 2 is expressly stated to be a sealed package, and Aizikowitz expressly defined classes and interfaces in a package to be non-public, as quoted above. Aizikowitz teaches that the operations are performed on the non-public classes and interfaces in the package while in Claim 3 such classes are excluded. Thus, Aizikowitz must be modified, and then that modification combined with the primary reference. The rejection cites only to a general knowledge of an object-orientated library to justify such a modification. The rejection has provided no basis for taking these teachings

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

concerning non-public classes and interfaces and using them in an entirely different context and in an entirely different process for removing redundant components from an object. Applicant requests reconsideration and withdrawal of the third obviousness rejection of Claim 3.

Claim 4 depends from Claim 3 and so distinguishes over the combination of references for at least the same reasons as Claim 3. Applicant requests reconsideration and withdrawal of the third obviousness rejection of Claim 4.

Independent Claims 10, 17, and 22 includes the limitations discussed above with respect to Claim 3 and so the comments with respect to Claim 3 are applicable and incorporated herein by reference. Applicant requests reconsideration and withdrawal of the third obviousness rejection of each of Claims 10, 17, and 22.

Claim 11 depends from Claim 10, Claim 18 depends from Claim 17, and Claim 23 depends from Claim 22, and so each of Claims 11, 18, and 23 distinguishes over the combination of references for at least the same reasons as the independent claim from which it depends. Applicant requests reconsideration and withdrawal of the third obviousness rejection of each of Claims 11, 18, and 23.

//
//
//
//
//
//
//
//
//
//
//

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

Claims 3, 4, 10, 11, 17, 18, and 22 to 35 remain in the application. Claims 3, 10, 17, 22, 24, 27, 30, and 33 are amended. For the foregoing reasons, Applicant(s) respectfully request allowance of all pending claims. If the Examiner has any questions relating to the above, the Examiner is respectfully requested to telephone the undersigned Attorney for Applicant(s).

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on May 16, 2006.

Attorney for Applicant(s) — May 16, 2006 Date of Signature

Respectfully submitted,

Forrest Gunnison
Attorney for Applicant(s)
Reg. No. 32,899
Tel.: (831) 655-0880

GUNNISON, McKAY & HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888